Alignment-based Trust for Resource Finding in Semantic P2P Networks

Manuel Atencia[1,2], Jérôme Euzenat[1],
Giuseppe Pirrò[3], and Marie-Christine Rousset[2]

[1] INRIA, Grenoble, France
{Manuel.Atencia,Jerome.Euzenat}@inrialpes.fr
[2] University of Grenoble, Grenoble, France
Marie-Christine.Rousset@imag.fr
[3] Free University of Bolzano-Bozen, Bolzano, Italy
giuseppe.pirro@unibz.it

Abstract. In a semantic P2P network, peers use separate ontologies and rely on alignments between their ontologies for translating queries. Nonetheless, alignments may be limited —unsound or incomplete— and generate flawed translations, leading to unsatisfactory answers. In this paper we present a trust mechanism that can assist peers to select those in the network that are better suited to answer their queries. The trust that a peer has towards another peer depends on a specific query and represents the probability that the latter peer will provide a satisfactory answer. In order to compute trust, we exploit both alignments and peers' direct experience, and perform Bayesian inference. We have implemented our technique and conducted an evaluation. Experimental results showed that trust values converge as more queries are sent and answers received. Furthermore, the use of trust improves both precision and recall.

1 Introduction

Peer-to-peer (P2P) systems have received considerable attention because their underlying infrastructure is very appropriate for scalable and flexible distributed applications over Internet. In P2P systems, there is no centralised control or hierarchical organisation: each peer is equivalent in functionality and cooperates with other peers in order to solve a collective task. P2P systems have evolved from simple keyword-based file sharing systems such as Napster and Gnutella to semantic data management systems such as Edutella [14], Piazza [8] or SomeWhere [1].

In this paper, by a *semantic P2P network* we refer to a fully decentralised overlay network of people or machines (peers) sharing and searching for resources (documents, videos, photos, data, services) based on their semantic annotations using ontologies. In semantic P2P systems, every peer is free to organise her local resources as instances of classes of her own ontology serving as query interface for other peers. Alignments between ontologies make possible to reformulate queries from one local peer vocabulary to another. The result of a query is a set of resources (*e.g.*, documents) which are instances of some classes corresponding, possibly via subsumption or equality, to the initial query posed to a specific peer.

Trust is widely acknowledged as a central factor when considering networks of autonomous interacting entities and notably in the context of the Semantic Web. When referring to the notion of trust, T. Berners-Lee advocates for a user to be able to search for reasons why he or she should be confident of a returned answer [3]. Trust is helpful to select, from a given set of peers, those that are expected to answer with most satisfactory instances. Peers may use this information for broadcasting their queries to a reduced set of peers and to have an approximation of the reliability of provided answers. Furthermore, peers may preventively send selected queries in order to improve the trust they have towards another peer. Finally, by identifying "weak correspondences", peers may signal faulty alignments and trigger new matching of the ontologies.

Several proposals have been made that do not share the same meaning for trust [15, 2]. Many are user/agent/peer centred and rely on the assumption that all peers share similar implicit goals. Trust is then closely related to the notion of reputation in a community.

In contrast, in the context of semantic P2P systems, each peer may have her own view on how categorising the resources that are exchanged between peers. For this reason, we rather promote the computation of subjective trust values based on direct experiences between peers. We also argue for a finer grained approach to trust in order to take into account the fact that, for answers provided by the same peer, the trust into these answers may vary according to which class they are instance of within the peer ontology.

An Illustrative Scenario

Consider a semantic P2P system for exchanging bookmarks, in which a peer *Alice* organises her bookmarks according to two main categories: *FavouriteMusic* and *GoodRestaurants*. These in turn are divided into subcategories: *Jazz*, *PopRock* and *Folk* for *FavouriteMusic*, and *Italian* and *Chinese* for *GoodRestaurants*. Within the Semantic Web, this can be implemented as a lightweight ontology that can be expressed in RDFS, in which categories and subcategories correspond to classes and subclasses, and the URLs identifying bookmarks correspond to URIs declared as instances of some classes.

Suppose that *Alice* is acquainted with *Bob* and *Chris* with whom she shares some interests in music and restaurants. This is captured by *correspondences* between her ontology and *Bob*'s and *Chris*'s ontologies. If *Bob* organises his best-of songs according to his favourite singers (*e.g.*, the classes *MichaelJackson* and *LouisArmstrong* are declared as subclasses of *BestSongs* in his ontology), the following correspondence expresses that any URL bookmarked by *Bob* as an instance of his class *MichaelJackson* can be bookmarked by *Alice* as an instance of her own class *PopRock*:[4]

$$Bob : MichaelJackson \leqslant Alice : PopRock$$

[4] We make use of the notation $P : A$ for identifying a class A of peer P's ontology.

An alignment between two peer ontologies is a set of correspondences between some classes used by these peers. Figure 1 shows the ontologies and alignments between *Alice*'s, *Bob*'s and *Chris*'s ontologies. It must be seen as a (small) part of a semantic P2P system that can be queried for resource finding.

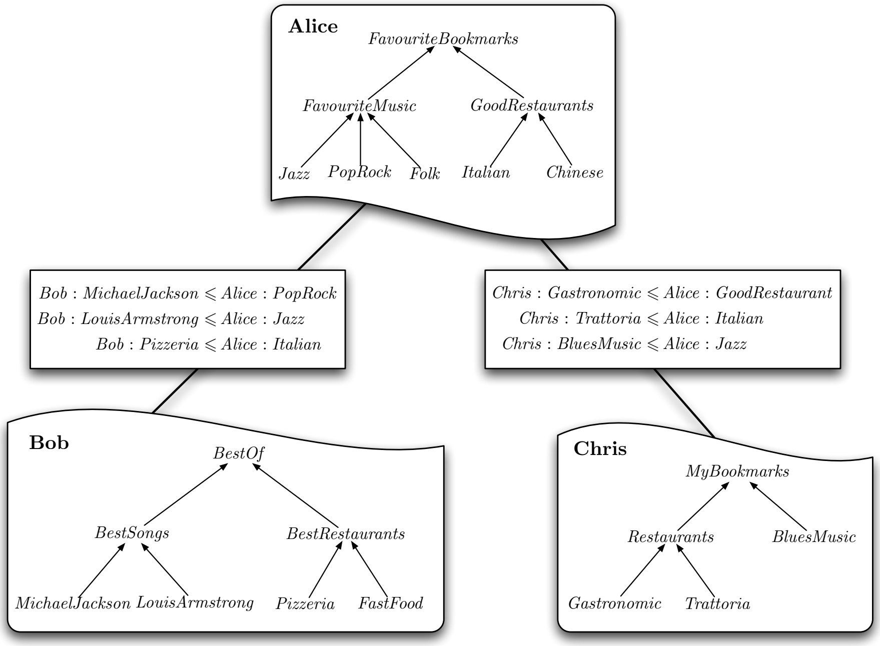

Alice
FavouriteBookmarks
FavouriteMusic
GoodRestaurants
Jazz
PopRock
Folk
Italian
Chinese
Bob : MichaelJackson ⩽ Alice : PopRock
Bob : LouisArmstrong ⩽ Alice : Jazz
Bob : Pizzeria ⩽ Alice : Italian
Chris : Gastronomic ⩽ Alice : GoodRestaurant
Chris : Trattoria ⩽ Alice : Italian
Chris : BluesMusic ⩽ Alice : Jazz
Bob
BestOf
BestSongs
BestRestaurants
MichaelJackson
LouisArmstrong
Pizzeria
FastFood
Chris
MyBookmarks
Restaurants
BluesMusic
Gastronomic
Trattoria

Fig. 1. Three semantic peers in a P2P semantic network.

Suppose *Alice* wants to get bookmarks from her acquaintances in the network to enrich her bookmarks about *Italian* restaurants. The alignments between her ontology and *Bob*'s and *Chris*'s ontologies allow to reformulate her initial query about *Italian* restaurants into the query *Pizzeria* asked to *Bob*, and *Trattoria* asked to *Chris*. As his answer set, *Bob* will return to her the set of instances in the extension of his class *Pizzeria*, and *Chris* the set of instances in the extension of his class *Trattoria*.

Alice notices that *Chris* has some bookmarks in common with her, and thus tends to trust *Chris* for providing her with instances that fits well with her taste in terms of *Italian* restaurants. Subsequently, she may be inclined to add new bookmarks in her class *Italian* when they come from *Chris*.

However, although she trusts *Chris* for restaurants, she may not trust him for his musical tastes. For instance, for getting new bookmarks about *Jazz*, she can discover by choosing a sample of the set of URLs returned by *Chris* as the extension of his class *BluesMusic* that very few corresponding music files fit well with her taste in terms of Jazz music. For music, she will later tend not to trust *Chris* and will prefer to query *Bob* on this topic.

Contributions

In this paper we propose a probabilistic model to handle trust in a semantic P2P setting. We define the trust of a peer P towards another peer P' regarding a class C (belonging to P's ontology) as the probability that an instance returned by P' as an answer to the query asking for instances of C is satisfactory for P. In order to compute trust, we exploit the information provided by peers' ontologies and alignments, along with the information that comes from direct experience. Trust values are refined over time as more queries are sent and answers received.

We have designed an experimental protocol to study the convergence of trust, and to measure the gain of using trust for resource finding in practice.

Finally, a by-product of our trust model is a probabilistic setting for resource finding, in which the instances returned as answer for a given query are associated with a probability. This is in line with the recent trends towards probabilistic databases [4].

The paper is organised as follows. The background of our work is presented firstly. Then we introduce the notion of probabilistic populated ontology and the definition of trust. Later we explain the computation of trust and update of probabilistic populated ontologies. We discuss experimental results, and finally give some concluding remarks.

2 Preliminaries

In this section the components of a semantic peer-to-peer network are presented: populated ontologies, alignments and acquaintance graphs. The kind of queries that we take into account is also described.

2.1 Ontologies and Populated Ontologies

We draw a distinction between the ontological structure and the instances used to populate it. We deal with lightweight ontologies: classes linked by means of a less-general-than relation and a disjointness relation.

Definition 1. *An* ontology *is a tuple $O = \langle C, \leqslant, \perp \rangle$ where C is a non-empty finite set of* class symbols*; $\leqslant$ is a partial order on C; $\perp$ is an irreflexive and symmetric relation on C; and for all $c, c', d, d' \in C$,*

$$\textit{if } c \perp d,\ c' \leqslant c \textit{ and } d' \leqslant d \textit{ then } c' \perp d'$$

A populated ontology is the result of adding instances to an ontology in accordance to the intended meaning of the two ontological relations.

Definition 2. *A* populated ontology *$\mathcal{O}$ is a tuple $\langle O, I, ext \rangle$, where O is an ontology, I is a set of* instances*, and ext is a function that maps each class c of O with a subset $ext(c)$ of I called the* extension *of c, in such a way that the family of class extensions covers I, and for all classes c, d the following hold:*

1. *if $c \leqslant d$ then $ext(c) \subseteq ext(d)$, and*
2. *if $c \perp d$ then $ext(c) \cap ext(d) = \emptyset$.*

2.2 Alignments

In an open and dynamic environment as a P2P network, the assumption of peers sharing the same ontology is not realistic. But if peers fall back on different ontologies, there must be a way to connect ontologies and translate queries so that their addressees are able to process them. Typically this is done by means of alignments —sets of correspondences between semantically related ontological entities— and finding alignments is what ontology matching is aimed at (see [7]).

A correspondence between two classes c and c' of two ontologies O and O', respectively, is usually defined as a tuple $\langle c, c', r\rangle$ with $r \in \{\leqslant, =, \geqslant, \perp\}$, where $c \leqslant c'$ (or $\langle c, c', \leqslant\rangle$) is read "$c$ is less general than c'", $c = c'$ is read "c is equal to c'", $c \geqslant c'$ is read "c is more general than c'", and $c \perp c'$ is read "c is disjoint from c'". Here, however, we deal with a more general notion of a correspondence inspired from [6].

Definition 3. *Let O and O' be two ontologies, and let c and c' be two classes of O and O', respectively. A* correspondence *between c and c' is a tuple $\langle c, c', R\rangle$ with $R \in 2^{\Gamma}$ where Γ is the set $\{=, >, <, \between, \perp\}$. An* alignment *$\mathcal{A}$ between O and O' is a set of correspondences between classes of O and O'.*

In such correspondences, a class is connected to another through a set of base relations to be thought of as an exclusive disjunction. For instance, $c\{>, <\}c'$ (*i.e.*, $\langle c, c', \{>, <\}\rangle$) is read "either c is more general than c' or less general than c'". In this way, we can express uncertainty with regard to the alignment relation. Note that the relations '$\geqslant$' and '$\leqslant$' can be seen as abbreviations for $\{=, >\}$ and $\{=, <\}$, respectively. Secondly, a nonstandard symbol '$\between$' is introduced. It reflects the idea of overlapping: classes the extensions of which share some instances but no one is equal to or contained into the other. Finally, $c\,\Gamma\,c'$ states total uncertainty about the relation between c and c'.

According to Definition 3, an alignment may include correspondences that link the same two classes through different relations, or no one connecting two particular classes. However, one would like alignments to relate any pair of classes and to do it in one way. If there exists no correspondence between c and c' in an alignment $\mathcal{A}$, we can simply add $\langle c, c', \Gamma\rangle$. If $\langle c, c', R\rangle, \langle c, c', S\rangle \in \mathcal{A}$ with $R \neq S$, we can replace both with $\langle c, c', R \cap S\rangle$. This follows the interpretation of alignments as a set of correspondences which all hold. The resulting alignment is said to be normalised.

Definition 4. *Let $\mathcal{A}$ be an alignment between two ontologies O and O'. The* normalisation *of $\mathcal{A}$ is the alignment $\overline{\mathcal{A}}$ made up of all correspondences $\langle c, c', R\rangle$ with $c \in C$, $c' \in C'$ and $R = \bigcap \mathcal{R}_{\mathcal{A}}(c, c')$ where $\mathcal{R}_{\mathcal{A}}(c, c') = \{S : \langle c, c', S\rangle \in \mathcal{A}\}$. The alignment $\mathcal{A}$ is said to be* normalised *providing $\mathcal{A} = \overline{\mathcal{A}}$.*

Remark 1. Recall that if $\mathcal{R}_{\mathcal{A}}(c, c') = \emptyset$ then $\bigcap R_{\mathcal{A}}(c, c') = \Gamma$.

All alignments considered in this work are assumed to be normalised.

2.3 Peers and Acquaintance Graphs

We consider a finite set $\mathcal{P} = \{P_i\}_{i=1}^n$ of peers. In this work, P_i will be identified by i. We assume that each peer P_i is associated with one populated ontology $\mathcal{O}_i = \langle O_i, I_i, ext_i \rangle$ (where $1 \leq i \leq n$).

An acquaintance graph stands for peers' acquaintances (or neighbours) in the network. As usual, a link between two peers reflects the fact that they know the existence of each other. In addition, we assume that there exists one alignment between their respective ontologies.

Definition 5. *An* acquaintance graph *is a labelled directed graph* $\langle \mathcal{P}, \textsc{acq} \rangle$ *where* $\mathcal{P} = \{P_i\}_{i=1}^n$ *is the set of vertices and any edge in* $\textsc{acq}$ *is of the form* $\langle i, j \rangle$ *with* $i \neq j$*, and it is labelled with an alignment* $\mathcal{A}_{ij}$ *between ontologies* O_i *and* O_j*. Moreover, if* $\langle i, j \rangle \in \textsc{acq}$ *then* $\langle j, i \rangle \in \textsc{acq}$ *and* $\mathcal{A}_{ji}$ *is the inverse of* $\mathcal{A}_{ij}$*.*[5]

Peer P_j *is said to be an* acquaintance *of peer* P_i *if* $\langle i, j \rangle \in \textsc{acq}$*. The set of acquaintances of* P_i *is denoted by* $\textsc{acq}(P_i)$*.*

Remark 2. Note that, given two ontologies O and O', we can always consider the trivial alignment, that is, the one that is made up of all correspondences $\langle c, c', \Gamma \rangle$ with $c \in C$ and $c' \in C'$.

2.4 Queries and Query Translations

Peers pose queries to obtain information concerning other peers' populated ontologies. We deal with a simple query language, as peers can only request class instances: if peer P_j is an acquaintance of peer P_i, she may be asked

$$\mathcal{Q} = c(X)? \tag{1}$$

by P_i with $c \in O_i$. Now, since we do not assume that all peers share the same ontology, queries may require to be translated for their recipients to be able to process them.

Query translations are determined by correspondences of the alignments of the network. Specifically, if peer P_i wants to send $\mathcal{Q}$ to P_j, she will first choose one correspondence $\langle c, d, R \rangle \in \mathcal{A}_{ij}$ (typically R is equal to '$=$' or '$>$') and then send P_j the translation

$$\mathcal{Q}' = d(X)? \tag{2}$$

The answer to (1) through its translation (2) is the set of instances of class d in P_j's populated ontology. Unlike queries, we assume that no translation of instances is ever required. Since alignments may be unsound and incomplete, this answer may contain *unsatisfactory* instances, *i.e.*, instances which are not considered instances of c by P_i.

A peer cannot foresee whether the answer that another peer provides to one of her queries contains satisfactory instances or not, but this uncertainty can be estimated with the help of a trust mechanism.

[5] Given an alignment $\mathcal{A}$ between O and O', the inverse of $\mathcal{A}$ is the alignment $\mathcal{A}^{-1} = \{\langle c', c, R^{-1} \rangle : \langle c, c', R \rangle \in \mathcal{A}\}$ between O' and O, where $R^{-1} = \{r^{-1} : r \in R\}$ and r^{-1} is '$>$' and '$<$' if r is '$<$' and '$>$', respectively, and $r^{-1} = r$ otherwise.

3 The Trust Mechanism

As mentioned above we look at trust as a way to estimate the proportion of satisfactory instances in a peer answer. The notion of satisfactory instance can be faithfully captured by an ideal populated ontology $\mathcal{O}_i^*$ that corresponds to a hypothetical situation in which peer P_i classified all instances of the network according to her ontology O_i. In this way we can express the fact that P_i considers an arbitrary instance a as an instance of $c \in O_i$ by $a \in ext_i^*(c)$. It is assumed that $O_i = O_i^*$ and $ext_i(c) \subseteq ext_i^*(c)$ for every class $c \in C_i$. This populated ontology is referred to as the *reference populated ontology* of peer P_i.

If peer P_i receives a set B as an answer to the query (2), the proportion of satisfactory instances is given by the conditional probability $p(ext_i^*(c)|B)$. The probability space under consideration here is the triple $(\Omega, \mathfrak{A}, p(\cdot))$ where Ω is the set of instances of the network (a finite set), the σ-algebra $\mathfrak{A}$ is the power set of Ω, and $p(\cdot)$ is Laplace's definition of probability. Our approach for trust aims at finding approximations to these conditional probabilities. Before the definition of trust we introduce the notion of a probabilistic populated ontology.

3.1 Probabilistic Populated Ontologies

Once an answer is received, it can be (partly) added or not to the extension of the queried class. In order to capture the evolution of class extensions in the network, we consider a time variable $t \in \mathbb{N}$, and we will write $\mathcal{O}_i^t$ to denote peer P_i's populated ontology at instant t (beginning with $\mathcal{O}_i$):

$$\mathcal{O}_i = \mathcal{O}_i^0, \mathcal{O}_i^1, \ldots, \mathcal{O}_i^t, \ldots \tag{3}$$

We assume that the underlying ontology never changes, *i.e.*, $O_i = O_i^t$ $(t \in \mathbb{N})$, and that the sequence of class extensions $\{ext_i^t(c)\}_{t \in \mathbb{N}}$ is monotonically increasing for all $c \in C_i$.

Nonetheless, since we deal with probabilities new instances may not be 100% satisfactory. For this reason, at $t \in \mathbb{N}$, peer P_i is associated with a *probabilistic populated ontology.*

Definition 6. *Peer P_i's probabilistic populated ontology at time t is a triple*

$$\widetilde{\mathcal{O}}_i^t = \langle O_i, I_i^t, \widetilde{ext}_i^t \rangle$$

where I_i^t is a set of instances and $\widetilde{ext}_i^t$ is a function that maps each class c of O_i with its probabilistic extension

$$\widetilde{ext}_i^t(c) = \langle A^*, \mathcal{F} \rangle \textit{ with } \mathcal{F} = \{\langle A^k, [p^k, q^k] \rangle\}_{k \in K} \textit{ where}$$

- *A^* is a (possibly empty) subset of $ext_i^*(c)$, that is, a set of instances which are certain to be instances of the class c, and all*
- *A^k are pairwise disjoint subsets of I_i^t which are also disjoint from A^*, and all $[p^k, q^k]$ are distinct subintervals of the unit interval* $[0, 1]$*, where $k \in K$ and K is a (possibly empty) index set of integers.*

Furthermore, the tuple $\mathcal{O}_i^t = \langle O_i, I_i^t, ext_i^t \rangle$ *with* $ext_i^t(c) = A^* \uplus \biguplus_{k \in K} A^k$ *must be a populated ontology (so that the axioms that relate classes with their extensions are fulfilled).*

A probabilistic extension $\widetilde{ext}_i^t(c)$ can be seen as a classical extension $ext_i^t(c)$ partitioned into a number of subsets $A^*, A^1, \ldots, A^n$. All instances of A^* are sure to be instances of the class c and then $p(ext_i^*(c)|A^*) = 1$. However, the set A^k $(1 \leq k \leq n)$ may contain instances that are actually not instances of c. The idea behind the interval $[p^k, q^k] \subseteq [0, 1]$ is that there exists some statistical evidence for $p^k \leq p(ext_i^*(c)|A^k) \leq q^k$.[6] The way probabilistic extensions are built is explained in Section 3.5.

Remark 3. Every populated ontology $\mathcal{O}_i$ can be seen as a probabilistic populated ontology $\widetilde{\mathcal{O}}_i = \langle O_i, I_i, \widetilde{ext}_i \rangle$ where $\widetilde{ext}_i(c) = \{\langle ext_i(c), \emptyset \rangle\}$ for all $c \in C_i$.

Peers build probabilistic populated ontologies as more queries are sent and answered (starting with the "probabilistic version" of $\mathcal{O}_i$):

$$\widetilde{\mathcal{O}}_i = \widetilde{\mathcal{O}}_i^0, \widetilde{\mathcal{O}}_i^1, \ldots, \widetilde{\mathcal{O}}_i^t, \ldots \tag{4}$$

And what was said about (3) at the beginning of this section holds for the underlying populated ontologies of (4).

3.2 Definition of Trust

With the new terminology, P_j's answer to query (1) via its translation (2) at time t is the extension $ext_j^t(d)$, and an arbitrary instance $a \in ext_j^t(d)$ is qualified as satisfactory provided that $a \in ext_i^*(c)$. The proportion of satisfactory instances in $ext_j^t(d)$ is given by the conditional probability $p(ext_i^*(c)|ext_j^t(d))$. Our proposal is that the higher this value is, the more P_i trusts P_j.

Definition 7. *Let us consider two peers* P_i *and* P_j $(i \neq j)$ *and two classes* c *and* d *of* O_i *and* O_j*, respectively. The* trust *that* P_i *has towards* P_j *with respect to the translation* $\langle c, d \rangle$ *at time* t *is the conditional probability* $p(ext_i^*(c)|ext_j^t(d))$ *and it is denoted by* $trust^t(P_i, P_j, \langle c, d \rangle)$*.*

This idea is slightly different from most of the existing approaches for trust. In our setting cheating is not directly addressed: unsatisfactory answers are seen as the result of peers' incapacity to understand each other. In addition, trust is dependent on translations: peers may be very trustworthy in regard with some translations but not with others. In the following section, we explain our approach for computing trust. It exploits the information provided by alignments and revises it with direct experience.

[6] The use of intervals follows Lukasiewicz's notation for conditional constraints in probabilistic knowledge bases [11].

3.3 Computation of Trust

Our approach for trust aims at approximating $trust^t(P_i, P_j, \langle c, d\rangle)$ by Bayesian inference. A probability distribution $T^t(P_i, P_j, \langle c, d\rangle)$ represents P_i's belief about $\theta = p(ext_i^*(c)|ext_j^t(d))$. If there is no direct experience, alignments are taken to construct prior beliefs. Answers are later used to revise these beliefs. As they can be of a size that cannot be processed manually, we propose to perform sampling with replacement in order to estimate the number of satisfactory instances. We work with beta distributions as they are typically used to describe the parameter of a binomial distribution.

No direct experience: alignment-based trust. If $T^t(P_i, P_j, \langle c, d\rangle)$ is not defined (this is the case when, for instance, $t = 0$), we fall back on alignments. Peers P_i and P_j's ontologies are linked through $\mathcal{A}_{ij}$. Since this alignment is normalised then there exists a unique $R \subseteq \Gamma$ such that $\langle c, d, R\rangle \in \mathcal{A}_{ij}$. The intending meaning of correspondences is

$$\begin{array}{lll} R = \{=\} & \text{iff} & ext_i^*(c) = ext_j^*(d) \\ R = \{>\} & \text{iff} & ext_i^*(c) \supset ext_j^*(d) \\ R = \{<\} & \text{iff} & ext_i^*(c) \subset ext_j^*(d) \\ R = \{\perp\} & \text{iff} & ext_i^*(c) \cap ext_j^*(d) = \emptyset \\ R = \{\between\} & \text{iff} & \text{none of the above holds} \end{array}$$

Hence, provided that $ext_j^t(d) \subseteq ext_j^*(d)$,

$$\begin{array}{rll} \text{if } R \text{ is `$=$' or `$>$'} & \text{then} & p(ext_i^*(c)|ext_j^t(d)) = 1 \\ \text{if } R \text{ is `$\perp$'} & \text{then} & p(ext_i^*(c)|ext_j^t(d)) = 0 \\ \text{if } R \text{ is `$<$' or `$\between$'} & \text{then} & p(ext_i^*(c)|ext_j^t(d)) \in [0, 1] \end{array}$$

In the general case, R is a set $\{r_1, \ldots, r_n\} \subseteq \Gamma$ (with $n \leq 5$). If we assume that all relations in R are equiprobable, by the law of total probability, we have

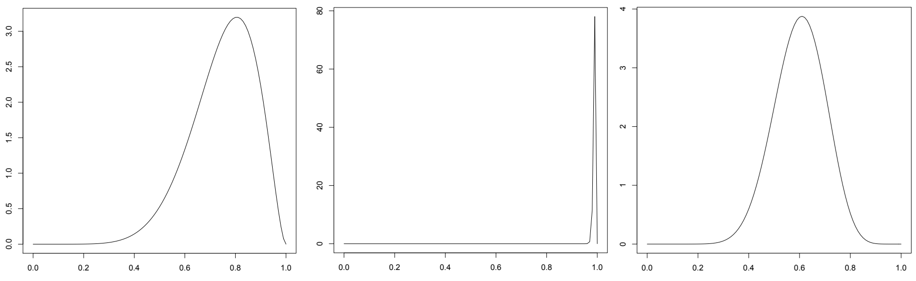
$$trust^t(P_i, P_j, \langle c, d\rangle) \in [u, v] \quad \text{with} \quad u = \frac{1}{n}\sum_{k=1}^{n} a_k \quad v = \frac{1}{n}\sum_{k=1}^{n} b_k$$

where $[a_k, b_k] = [1, 1]$ if r_k is '$=$' or '$>$', $[a_k, b_k] = [0, 0]$ if r_k is '$\perp$', and finally $[a_k, b_k] = [0, 1]$ if r_k is '$<$' or '$\between$' ($k = 1, \ldots, n$).

The information above can be used to construct P_i's prior belief about the parameter $\theta = p(ext_i^*(c)|ext_j^t(d))$ by means of a beta distribution $\text{Beta}(\alpha, \beta)$. If $[u, v] = [0, 1]$, we take the uniform distribution $U[0, 1] = \text{Beta}(1, 1)$. If not, and $u < v$, we equal $[u, v]$ with $[\mu - 2\sigma, \mu + 2\sigma]$ and then find $\text{Beta}(\alpha, \beta)$ whose mean and deviation are μ and σ. This is the standard way to find a confidence interval based on the normal distribution. If $u = v$, we proceed with $\sigma = 0.005$ and $\mu = u$ unless $u = 1$ and $u = 0$, in which cases $\mu = 0.99$ and $\mu = 0.01$, respectively.[7] In this way, we define the trust distribution $T^t(P_i, P_j, \langle c, d\rangle) = \text{Beta}(\alpha, \beta)$.

[7] The values for α and β can be found by solving $\mu = \frac{\alpha}{\alpha+\beta}$ and $\sigma = \sqrt{\frac{\mu(\mu-1)}{\alpha+\beta+1}}$.

Example 1. If $R = \{<, =\}$ then $[u, v] = [.5, 1]$. If we make $[.5, 1] = [\mu - 2\sigma, \mu + 2\sigma]$ then $\mu = .75$ and $\sigma = .125$. This leads to Beta$(8.25, 2.75)$ whose shape is depicted in Figure 2(a). Figure 2 is completed with the shapes of beta distributions for the relations $R = \{=\}$ and $R = \Gamma$. The latter corresponds to Beta$(0.4, 0.8)$.[8]



(a) $\langle c, \{<, =\}, d\rangle$ (b) $\langle c, \{=\}, d\rangle$ (c) $\langle c, \Gamma, d\rangle$

Fig. 2. Beta distributions for different correspondences.

Direct experience. Trust at time t is used to choose a peer to which to send a query, as well as a class through which to translate it. This is explained in detail in Section 3.4. Let us imagine that P_i receives $B = ext_j^t(d)$. A sampling with replacement is performed over B in order to estimate the number of satisfactory instances. Let $S \subseteq B$ be a sample (strictly speaking, S is a multiset). We assume that every peer can call an *oracle* (typically the user) to find out whether an instance is satisfactory or not. More specifically, given $a \in S$, P_i's oracle provides a yes/no response to the question: "$a \in ext_i^*(c)$?". Even this, nonetheless, may be a high burden for P_i's oracle. We can benefit from peers' populated ontologies to process some instances automatically without the need to call oracles. Recall that P_i is associated with a probabilistic populated ontology $\widetilde{\mathcal{O}}_i^t$, and that the probabilistic extension of class c includes a set A^* of instances which are certain to be instances of $ext_i^*(c)$. So if $a \in B \cap A^*$, $a \in ext_i^*(c)$. We can also identify unsatisfactory instances automatically: if $a \in S$ is such that there exists c' in O_i with $c \perp c'$ and $a \in A'^*$ then $a \notin ext_i^*(c)$. The remaining instances are processed by peer P_i's oracle.

Assume that $T^t(P_i, P_j, \langle c, d\rangle) = \text{Beta}(\alpha, \beta)$. If s is the sample size, s^+ is the number of successes (satisfactory instances), and $s^- = s - s^+$ is the number of failures, peer P_i's posterior belief about $\theta = p(ext_i^*(c)|ext_j^t(d))$ is summarised in Beta$(\alpha + s^+, \beta + s^-)$. Thus we define

$$T^{t+1}(P_i, P_j, \langle c, d\rangle) = \text{Beta}(\alpha + s^+, \beta + s^-) \tag{5}$$

[8] Although $c\Gamma d$ stands for total uncertainty about the relation between c and d, the mean of its associated beta distribution, Beta$(0.4, 0.8)$, is not 0.5 but 0.6. However, our aim is not to find out the correct relation between c and d, but to estimate the probability $p(ext_i^*(c)|ext_j^t(d))$. In this sense, total uncertainty arises with $c < d$, $c \between d$ or $c\{<, \between\}d$, which are all modelled with a uniform distribution (whose mean is 0.5).

3.4 Use of Trust

Imagine that peer P_i wants to query $c(X)$? ($c \in C_i$) at time $t \in \mathbb{N}$. Then P_i chooses an element from the set

$$\mathcal{P}_0 = \{\langle P_j, d_j\rangle : P_j \in \textsc{acq}(P_i) \text{ and } d_j \in O_j\}$$

so that, if $\langle P_{j_0}, d_{j_0}\rangle$ is the preferred tuple, P_i will send $d_{j_0}(X)$? to P_{j_0}. This choice depends on trust: P_i opts for $\langle P_{j_0}, d_{j_0}\rangle$ iff

$$\mathrm{E}(T^t(P_i, P_{j_0}, \langle c, d_{j_0}\rangle)) = \max_{\langle P_j, d_j\rangle \in \mathcal{P}_0} \{\mathrm{E}(T^t(P_i, P_j, \langle c, d_j\rangle))\}$$

where $\mathrm{E}(\cdot)$ denotes the expected value of a distribution.

3.5 Updating Probabilistic Populated Ontologies

In the end, trust is used for class extensions to be increased with new satisfactory instances. If peer P_i receives $B = ext_j^t(d)$ as an answer to "$c(X)$?" then B will be (partly) added to $ext_i^t(c)$. In line with the computation of trust based on direct experience (see Section 3.3), the set B is partitioned into three subsets:

$$B = B_{aut}^+ \uplus B_{aut}^- \uplus B_{\overline{aut}}$$

- $B_{aut}^+ = \{a \in B : a \in ext_i^t(c)\} = B \cap ext_i^t(c)$
- $B_{aut}^- = \{a \in B : \text{there exists } c' \in O_i \text{ with } a \in ext_i^t(c') \text{ and } c \perp c'\}$
- $B_{\overline{aut}} = \{a \in B : a \notin B_{aut}^+ \text{ and } a \notin B_{aut}^-\} = B \setminus (B_{aut}^+ \uplus B_{aut}^-)$

The set B_{aut}^+ contains the instances in B that already belong to $ext_i^t(c)$, and B_{aut}^- comprises those instances that, if added to $ext_i^t(c)$, would yield to a logical inconsistency. The set $B_{\overline{aut}}$ embodies the new information that can be included in $ext_i^t(c)$.[9] Since the answer B was received as the result of a comparison of trust values, it seems reasonable to add all instances of $B_{\overline{aut}}$ to $ext_i^t(c)$. The fact that these instances may not be 100% satisfactory, though, should be reflected in P_i's populated ontology. As described in Section 3.1, probabilistic populated ontologies are designed for this purpose.

The set $B_{\overline{aut}}$ will be included in $ext_i^t(c)$ along with an interval $[p, q]$ such that $p \leq p(ext_i^*(c)|B_{\overline{aut}}) \leq q$ on the basis of statistical evidence. Again, we propose to perform Bayesian inference, but, instead of weighing more on P_i's oracle, we lean on the previous sampling and make use of the formula

$$p(ext_i^*(c), B_{\overline{aut}}|ext_j^t(d)) = p(ext_i^*(c)|B_{\overline{aut}}) \cdot p(B_{\overline{aut}}|ext_j^t(d)) \tag{6}$$

Let us explain this in detail. The probability $p(B_{\overline{aut}}|ext_j^t(d))$ represents the proportion of instances of $B_{\overline{aut}}$ in $ext_j^t(d)$ and its computation is straightforward. By monotonicity, we have

$$0 \leq p(ext_i^*(c), B_{\overline{aut}}|ext_j^t(d)) \leq p(B_{\overline{aut}}|ext_j^t(d))$$

[9] The subscript "aut" stands for "automatic", as both instances from B_{aut}^+ and B_{aut}^- can be automatically processed, whereas this is not the case for $B_{\overline{aut}}$.

In order to compute a prior about $\vartheta = p(ext_i^*(c), B_{\overline{aut}} | ext_j^t(d))$, we proceed as the computation of alignment-based trust (Section 3.3): we equate the interval $[u, v]$, where $u = 0$ and $v = p(B_{\overline{aut}} | ext_j^t(d))$, with $[\mu - 2\sigma, \mu + 2\sigma]$, and then find $\mathrm{Beta}(\alpha, \beta)$ whose mean and deviation are μ and σ, respectively. A posterior is computed with the same sampling S used for Equation 5, but this time we count as a success any satisfactory instance that also belongs to $B_{\overline{aut}}$.

Let $\mathrm{Beta}(\alpha', \beta')$ be the resulting posterior, and let μ' and σ' be its mean and deviation. The set $B_{\overline{aut}}$ is included in $ext_i^t(c)$ along with the interval $[p, q]$ where

$$p = \frac{1}{p(B_{\overline{aut}} | ext_j^t(d))}(\mu' - 2\sigma') \quad q = \frac{1}{p(B_{\overline{aut}} | ext_j^t(d))}(\mu' + 2\sigma')$$

Hence, $p \leq p(ext_i^*(c) | B_{\overline{aut}}) \leq q$ with 95% probability, which is based on the normal approximation to the posterior density for ϑ and Equation 6. Actually, if S^+ denotes the set of satisfactory instances in the sample S, $B_{\overline{aut}}$ is partitioned into $B_{\overline{aut}} \cap S^+$ and $B_{\overline{aut}} \setminus S^+$, which are added to $ext_i^t(c)$ separately. Thus $[p, q]$ must be resized accordingly, and then replaced by another interval $[p', q']$. Below we explain explicitly how probabilistic populated ontologies are built.

As remarked in Section 3.1, $\widetilde{\mathcal{O}}_i^0$ is defined as the probabilistic version of P_i's initial populated ontology $\mathcal{O}_i$, that is,

- $\widetilde{\mathcal{O}}_i^0 = \widetilde{\mathcal{O}}_i$, and
- at time $t \in \mathbb{N}$, if $\widetilde{ext}_i^t(c) = \langle A^*, \mathcal{F} \rangle$ then we define

$$\widetilde{ext}_i^{t+1}(c) = \langle A^* \uplus (B_{\overline{aut}} \cap S^+), \mathcal{F} \uplus \langle B_{\overline{aut}} \setminus S^+, [p', q'] \rangle \rangle$$

In order for $\widetilde{\mathcal{O}}_i^{t+1}$ to be a probabilistic populated ontology, though, $B_{\overline{aut}}$ must be included in the extension of any superclass c' of c. For the sake of space, we give a brief explanation of how this is done. Notice first that no instance in $B_{\overline{aut}}$ belongs to the extension of a class disjoint from c' as $B_{aut}^- \cap B_{\overline{aut}} = \emptyset$ and c is a subclass of c'. All instances in $B_{\overline{aut}} \cap S^+$ are certainly instances of c' since $ext_i^*(c) \subseteq ext_i^*(c')$. Instead of $B_{\overline{aut}} \setminus S^+$ we include $B_{\overline{aut}} \setminus (S^+ \cup ext_i^t(c'))$ as some instances of $B_{\overline{aut}} \setminus S^+$ may already belong to $ext_i^t(c')$. In order to find an interval with which to estimate $p(ext_i^*(c') | B_{\overline{aut}} \setminus (S^+ \cup ext_i^t(c')))$, we proceed as before to approximate $p(ext_i^*(c) | B_{\overline{aut}} \setminus (S^+ \cup ext_i^t(c')))$ and then apply the monotonicity of probability. In this way, the upper bound that we obtain is equal to 1.

By construction, $\widetilde{\mathcal{O}}_i^{t+1}$ is a probabilistic populated ontology.

4 Experimental Analysis

This section reports on a preliminary experimental campaign that has been conducted to test the viability of the trust mechanism described in this paper.

We set out to answer two research questions:

1. Do trust values converge as more queries are sent and answers received?
2. Is there any gain in query-answering performance —measured in precision and recall— by using the trust technique?

In what follows we first describe the experimental setting and then explain the execution and evaluation.

4.1 Experimental Setting

The trust mechanism presented in this work has been implemented in a simulator written in Java. The simulator also deals with aspects indirectly related to trust, such as generation of P2P networks, populated ontologies and alignments. In the remainder of the section we elaborate more on these aspects.

P2P network topology. Social networks are well-known to exhibit small-world characteristics [5]. For this reason, a small-world topology was used for the entire evaluation. To generate this topology, we ran Kleinberg's algorithm included in the JUNG Java library.[10] A node in the network represents a peer associated with a populated ontology. The total number of peers in our evaluation was 20.

Populated ontologies. All populated ontologies in the evaluation had the same underlying ontology $O_i = O$. More specifically, we chose the ontological scheme described in [10] (with 64 classes). The semantic heterogeneity was reproduced by the way classes were populated with instances. The simulator implements an ontology population module which was utilised for both reference populated ontologies $\mathcal{O}_i^*$ and initial populated ontologies $\mathcal{O}_i = \mathcal{O}_i^0$. First, a set S of abstract instances is generated. In our evaluation, the size of S was 6000. Second, for each peer P_i, a sample S_i is taken from S. Furthermore, this sampling is performed in a way that S_i and S_j overlap for each pair i, j. The size of each S_i is determined with a Zipfian distribution, which is often used to approximate data in physical and social sciences [12]. The skewing factor considered was 0.5. Third, the top class of $\mathcal{O}_i^*$ is populated with S_i and a top-down population process is carried out by removing instances randomly for the remainder of classes. During this process, we check that all ontological axioms —subclass and disjoint relations— are fulfilled. Initial populated ontologies are generated in a similar way, starting this time with a sample of S_i instead of S to populate the top class in $\mathcal{O}_i$.

Alignment generation. A connection between peers P_i and P_j in the network (edge between nodes) is labelled with an alignment $\mathcal{A}_{ij}$ between their respective ontologies. This is seen as a declined version of a reference alignment $\mathcal{A}_{ij}^*$ which is never available to the peers. Thus we can capture the real practice of ontology matching. Reference alignments are built by comparing class extensions in the reference populated ontologies (for instance, $c < d$ is included in $\mathcal{A}_{ij}^*$ iff $ext_i^*(c) \subset ext_j^*(d)$). To build initial alignments, correspondences in reference alignments are discarded or replaced randomly in accord with global values for precision and recall. In our evaluation, we chose 0.6 for both measures.

4.2 Execution and Evaluation

From all peers and classes in the network we chose a subset $\mathcal{P}_0 \subseteq \mathcal{P}$ of 15 peers and a subset $C_0 \subseteq C$ of 25 classes randomly and ran 100 simulations. At each round $n \leq 100$ of the execution, a peer $P_i \in \mathcal{P}_0$ and a class $c \in C_0$ are randomly chosen. Then an acquaintance P_j of P_i and a class $d \in C_j$ are selected by using

[10] http://jung.sourceforge.net

the trust mechanism (Section 3.4). Notice that $C_i = C_j = C$ as we chose a single ontological scheme O. To process answers, the maximum number of oracle calls allowed was 40. The subset $B_{\overline{aut}} \subseteq ext_j^t(d)$ is included in peer P_i's probabilistic populated ontology if the expected value $\mathrm{E}(T^n(P_i, P_j, \langle c, d\rangle))$ is greater than a given threshold. In our evaluation, this threshold was 0.6.

In order to test the convergence of trust, we analysed the difference

$$\Delta^n = |\mathrm{E}(T^n(P_i, P_j, \langle c, d\rangle)) - p(ext_i^*(c)|ext_j^*(d))|$$

over the 10 most occurred queries. Figure 3 shows the experimentation results. After a number of rounds, Δ^n approached 0. Actually, in most of the cases, no more than 5 rounds were needed for Δ^n to be close to 0.1.

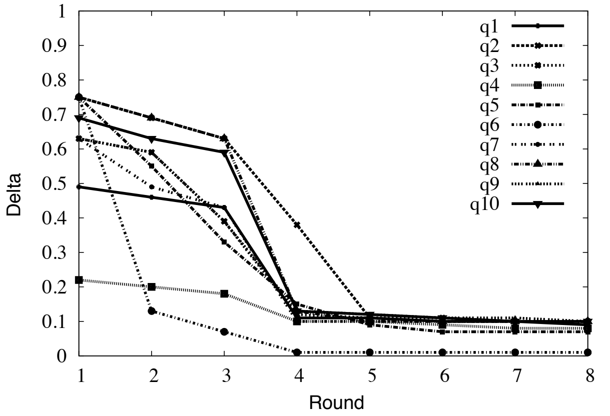


Fig. 3. Test of convergence of trust.

In order to test the gain in query-answering performance, we compared the use of the trust mechanism with a naive strategy. In the latter, peers randomly choose acquaintances and always accept their answers. For the evaluation to be fair, the same set of queries was used in both strategies. This time we analysed precision and recall measured by

$$Precision(n) = \frac{|ext_i^*(c) \cap ext_i^n(c)|}{|ext_i^n(c)|} \qquad Recall(n) = \frac{|ext_i^*(c) \cap ext_i^n(c)|}{|ext_i^*(c)|}$$

Figure 4 depicts the average precision and recall over the 100 rounds for the 20 most occurred queries. As expected, the naive strategy produced lower values for both measures. Furthermore, the use of the trust mechanism ensured high precision. However, this was not the case for recall. The reason is that peers only ask their neighbours, and these ones never change. As instances are spread all over the network, many instances may be unaccessible to peers. It is expected that if instances were more accessible, recall would be higher, but this remains to be experimented. The theoretical model presented in this paper is general enough to cover the case where peers receive answers from non-neighbour peers.

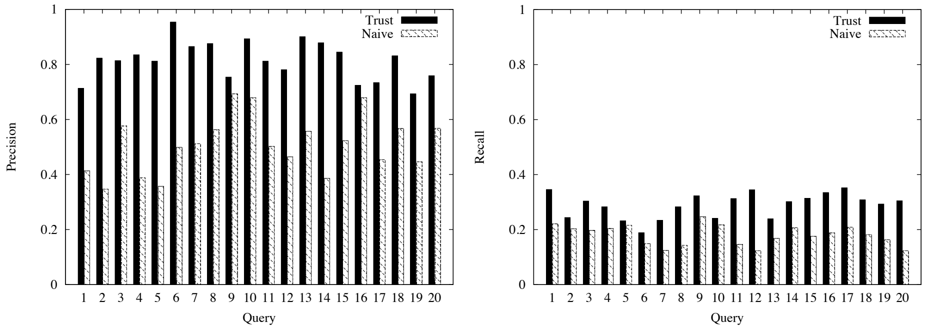


Fig. 4. Comparison between the use of trust and the naive strategy.

5 Concluding Remarks

We have proposed a trust mechanism in semantic P2P systems. The trust that a peer has towards another peer depends on a specific query and represents the probability that the latter will provide a satisfactory answer. In order to compute trust, we exploit alignments and peers' direct experience, and perform Bayesian inference. Preliminary experimental results show that trust values converge as more queries are sent and answers received, and that there is a gain in query-answering precision and recall when peers make use of the trust mechanism.

The notion of probabilistic populated ontology has been introduced. This is a by-product of trust computation that allows to store and process the instances obtained from query answers in the same way as it is done in probabilistic databases [4]. More precisely, a probabilistic populated ontology can be seen as a probabilistic database in which each fact $C(i)$ is associated with a (lower bound of) probability. As a result, query answers can be ranked, and only top-k answers can be returned to interested users. In addition, since trust evolves over time as more queries are spread over the network and their answers are processed and stored with their probabilities, the resulting probabilistic populated ontologies somehow *capture* and *compile* the results of a trust propagation.

Many different probabilistic approaches to trust can be found in the literature [16, 13]. Some also perform Bayesian inference over feedback on past interactions. However, to the best of our knowledge, our model is the only one which explicitly benefits from ontological content and alignments.

EigenTrust [9] is a peer-to-peer algorithm which, like ours, has a direct trust computation. Direct trust is then propagated among peers and aggregated to calculate global trust which can be very costly. As remarked above, we avoid this computation by exploiting the information on global trust stored and compiled in the probabilistic populated ontologies of acquaintance peers.

As future work, we plan to extend our trust model in order to deal with more expressive ontology and query languages. Although witness peers are not considered in this paper, the use of witness information is another future research line. Witness peers can help to find new trustworthy acquaintances. In this way, recall values can increase. Furthermore, the impact of malicious peers that hide or bias information, or lie, will be studied too.

Regarding the experimentation, we aim to perform a thorough experimental analysis concerning different network configurations in terms of number of peers, instances and oracle calls. Moreover, we want to investigate the relation between the quality of alignments and the speed of convergence of trust values.

Acknowledgements. This work is supported under the Dataring project, which is sponsored by the Agence Nationale de Recherche (ANR-08-VERS-007), and the Webdam project, sponsored by the European Research Council (FP7-226513).

References

1. Adjiman, P., Chatalic, P., Goasdoué, F., Rousset, M.C., Simon, L.: Distributed reasoning in a peer-to-peer setting: application to the Semantic Web. Journal of Artificial Intelligence Research 25, 269–314 (2006)
2. Artz, D., Gil, Y.: A survey of trust in computer science and the Semantic Web. Journal of Web Semantics 5(2), 58–71 (2007)
3. Berners-Lee, T.: Cleaning up the user interface. http://www.w3.org/DesignIssues/UI.html (1997)
4. Dalvi, N., Re, C., Suciu, D.: Query evaluation on probabilistic databases. IEEE Data Engineering Bulletin 29(1), 25–31 (2006)
5. Duncan J, W.: Networks, dynamics, and the small-world phenomenon. American Journal of Sociology 105(2), 493–527 (1999)
6. Euzenat, J.: Algebras of ontology alignment relations. In: Proceedings of the 7th International Semantic Web Conference, ISWC 2008. pp. 387–402 (2008)
7. Euzenat, J., Shvaiko, P.: Ontology matching. Springer (2007)
8. Halevy, A., Ives, Z., Tatarinov, I., Mork, P.: Piazza: data management infrastructure for Semantic Web applications. In: Proceedings of the 12th International Conference on World Wide Web, WWW 2003. pp. 556–567 (2003)
9. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The EigenTrust algorithm for reputation management in P2P networks. In: Proceedings of the 12th International Conference on World Wide Web, WWW 2003. pp. 640–651 (2003)
10. Lorenz, B.: Ontology of transportation networks. REWERSE project, IST-2004-506779, EU FP6 Network of Excellence (NoE). Deliverable (2005)
11. Lukasiewicz, T., Straccia, U.: Managing uncertainty and vagueness in description logics for the Semantic Web. Journal of Web Semantics 6(4), 291–308 (2008)
12. Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. The MIT Press (1999)
13. Mui, L., Mohtashemi, M., Ang, C., Szolovits, P., Halberstadt, A.: Ratings in distributed systems: a bayesian approach. In: Proccedings of the 11th Workshop on Information Technologies and Systems, WITS 2001 (2001)
14. Nejdl, W., Wolf, B., Qu, C., Decker, S., Sintek, M., Naeve, A., Nilsson, M., Palmér, M., Risch, T.: EDUTELLA: a P2P networking infrastructure based on RDF. In: Proceedings of the 11th International Conference on the World Wide Web, WWW 2002. pp. 604–615 (2002)
15. Sabater, J., Sierra, C.: Review on computational trust and reputation models. AI Review 24(1), 33–60 (2005)
16. Schillo, M., Funk, P., Rovatsos, M.: Using trust for detecting deceitful agents in artificial societies. Applied Artificial Intelligence 14(8), 825–848 (2000)